

Homework presentation

Rapid Web Development with MongoDB and Node.js

The Bioinformatics Lab SS 2013 - Wiki topic 12

Tikira Temu

18. June 2013

Mongodb

- install MongoDB

command line

```
sudo apt-get install mongodb
```

- Get familiar with the shell (Tutorial [1])
- Add two documents to the collection 'students'

mongoDB shell

```
manu= name: 'manuel', age: 24, courses: ['programming', 'math']  
tiki= name: 'tikira', age: 25, courses: ['informatics', 'neuroscience']  
use biolab  
db.students.insert(manu)  
db.students.insert(tiki)
```

NodeJS

■ install NodeJS

command line

```
sudo apt-get install python g++ make checkinstall  
mkdir /src && cd $_  
wget -N http://nodejs.org/dist/node-latest.tar.gz  
tar xzvf node-latest.tar.gz && cd node-v*  
./configure  
checkinstall #(remove the "v" in front of the version number)  
sudo dpkg -i node_*
```

Install the modules express and mongoose

■ install modules

command line

```
sudo npm install express
```

```
sudo npm install mongoose
```

■ check installations

```
1 var express = require('express');
2 var app = express();
3
4 app.get('/', function(req, res){
5     res.send('hello world');
6 });
7
8 app.listen(3000);
```



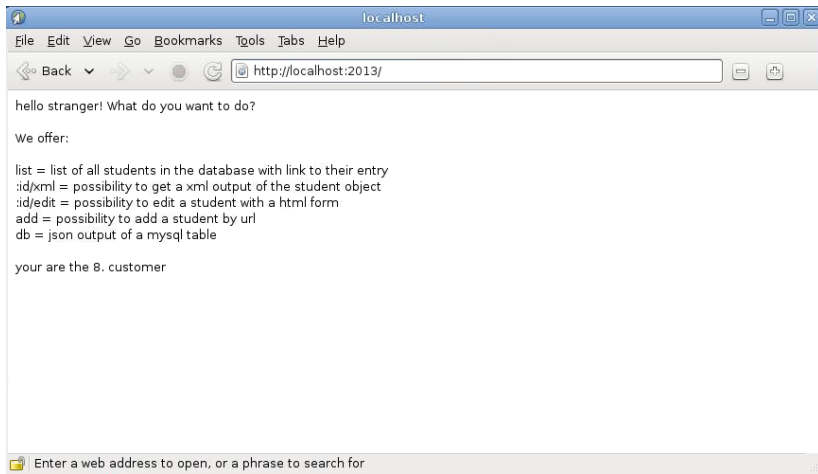
Define a mongoose model

```
1  var express = require('express');
2  var app = express();
3  app.listen(2013);
4
5  var mongoose = require('mongoose');
6  mongoose.connect('mongodb://localhost');
7  Schema = mongoose.Schema;
8
9
10 //create model
11 var Student = mongoose.model('student', new Schema ({
12   name: String,
13   age: Number,
14   courses: [String]
15 }));
16
17
18 //create documents
19 var manu = new Student ({
20   name: 'manuel',
21   age: 24,
22   courses: ['programming', 'math']
23 });
24
25 var tiki = new Student ({
26   name: 'tikira',
27   age: 25,
28   courses: ['informatics', 'neuroscience']
29 });
```

Define a mongoose model

```
32 //add documents
33 Student.create(manu, function(err,obj){
34     if(!err){
35         console.log("student " + manu.name + " created!");
36     }else{
37         console.log("Error: could not save student " + manu.name + "!");
38     }
39 });
40
41
42 Student.create(tiki, function(err,obj){
43     if(!err){
44         console.log("student " + tiki.name + " created!");
45     }else{
46         console.log("Error: could not save student " + tiki.name + "!");
47     }
48 });
49
50
```

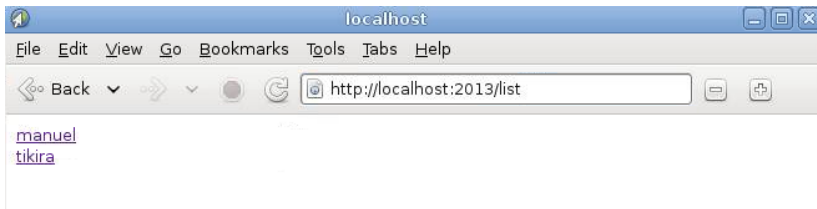
Create event bindings with express for the urls



Create event bindings with express for the urls

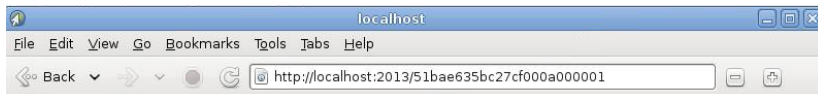
```
var counter = 0;
app.get('/',function(req,res){
  counter++;
  res.send('hello stranger! What do you want to do?'
    + '<br><br> We offer:'
    + '<br><br> list = list of all students in the database with link to their entry'
    + '<br> :id/xml = possibility to get a xml output of the student object'
    + '<br> :id/edit = possibility to edit a student with a html form'
    + '<br> add = possibility to add a student by url'
    + '<br> db = json output of a mysql table'
    + '<br><br> you are the ' + counter + 'th customer'
  );
});
```

GET /list



```
app.get('/list',function(req,res){
  Student.find(function (err,studs){
    var output="";
    for (var i = 0; i < studs.length; i++) {
      var curr = studs[i];
      output+='<a href="/' + curr.id + '"'>' + curr.name + '</a><br>';
    }
    res.send(output);
  });
});
```

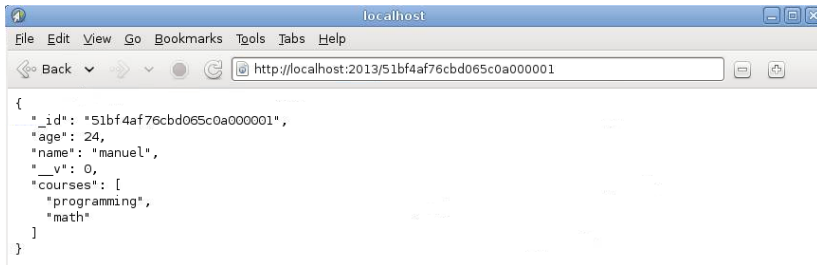
GET /:id



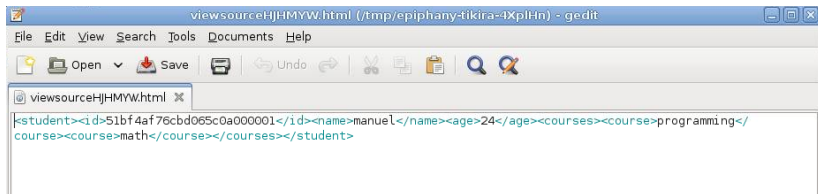
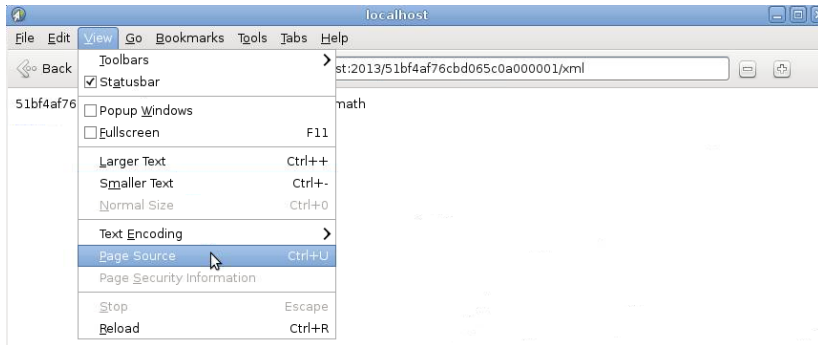
```
{
  "_id": "51bae635bc27cf000a000001",
  "age": 24,
  "name": "manuel",
  "__v": 0,
  "courses": [
    "programming",
    "math"
  ]
}
```

```
app.get('/:id', function(req, res) {
  Student.findById(req.params.id, function(err, stud) {
    res.send(stud);
  });
});
```

GET /:id/xml



GET /:id/xml



GET /:id/xml

Command line (module installation)

```
sudo npm install jsontoxml
```

```
var jsontoxml = require('jsontoxml');

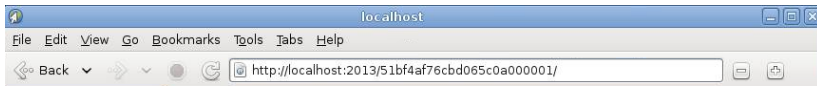
app.get('/:id/xml',function(req,res){
  Student.findById(req.params.id, function(err, stud) {
    res.send(jsontoxml({ student: {
      id: stud._id.toString(),
      name: stud.name,
      age: stud.age,
      courses: stud.courses.map(function(c){
        return {course:c};
      })
    } })));
  });
});
```

GET /:id/edit

A screenshot of a web browser window titled 'localhost'. The address bar shows the URL 'http://localhost:2013/51bf4af76cbd065c0a000001/edit'. The page contains a form with three input fields: 'name' with the value 'manuel', 'age' with the value '24', and 'courses' with the value 'programming,math'. Below these fields is an 'edit' button.

A second screenshot of the same web browser window. The 'age' input field now contains the value '25'. A mouse cursor is hovering over the 'edit' button. All other elements, including the URL and the 'name' and 'courses' fields, remain the same as in the first screenshot.

GET /:id/edit

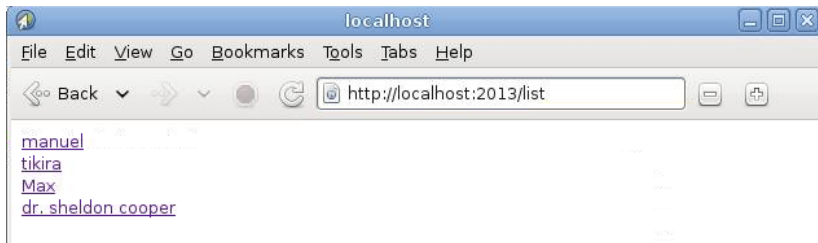


```
{
  "_v": 0,
  "_id": "51bf4af76cbd065c0a000001",
  "age": 25,
  "name": "manuel",
  "visitors": 1,
  "courses": [
    "programming,math"
  ]
}
```

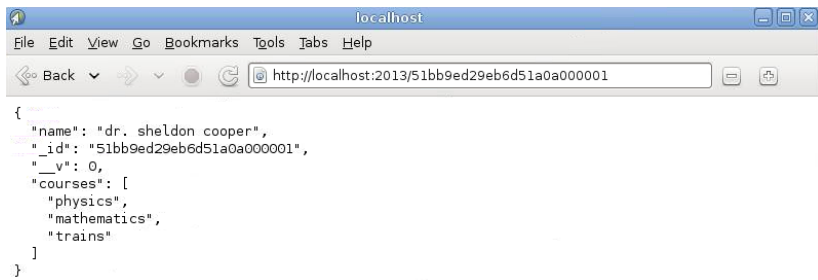
```
app.get('/:id/edit',function(req,res){
  Student.findById(req.params.id, function(err, stud) {
    res.write( '<html>');
    res.write( '<body>');
    res.write( '<form method="get">');
    res.write( '<p> <label> name: </label> <input type="text" name="name" value="'+stud.name+'"/></p>');
    res.write( '<p> <label> age: </label> <input type="text" name="age" value="'+stud.age+'"/></p>');
    res.write( '<p> <label> courses: </label> <input type="text" name="courses" value="'+stud.courses+'"/></p>');
    res.write( '<p><input type="submit" value="edit"/></p>');
    res.write( '</form>');
    res.write( '</body>');
    res.write( '</html>');
    res.end();

    Student.findByIdAndUpdate(req.params.id, req.query, function(err, obj) {
    });
  });
});
```

GET /add



GET /add



```
app.get('/add',function(req,res){
  Student.create(req.query, function(err,obj){
    res.redirect("/");
    res.send("New student created!");
  });
});
```

GET /db



GET /db

Command line (module installation)

```
sudo npm install mysql@2.0.0-alpha8
```

```
var mysql = require('mysql');
var connection = mysql.createConnection({
  //host: 'localhost',
  user: 'root',
  password: '...',
  database: 'db1'
});
connection.connect();

app.get('/db',function(req,res){
  connection.query('SELECT * FROM db1.testTable WHERE key1=23523;', function (err, rows, field){
    if(err){ throw err;}
    else{res.send(rows);}
  });
});
```

Add a field visitors in the Student model and count each time the `/:id` page of a student is requested

```
var Student = mongoose.model('student', new Schema ({
  name: String,
  age: Number,
  courses: [String],
  visitors: Number
}));

app.get('/:id', function(req, res) {
  Student.findById(req.params.id, function(err, stud) {
    var visits = stud.visitors + 1;
    Student.findByIdAndUpdate(req.params.id, { visitors: visits });
    res.send(stud);
  });
});
```

References

- [1] MongoDB Tutorial
- [2] jsontoxml
- [3] module: jstoxml
- [4] module: felixge / node-mysql



for your attention!